МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Искусственные нейронные сети»

Тема: «Регрессионная модель изменения цен на дома в Бостоне»

Студент гр. 7381 _____ Лукашев Р.С.

Преподаватель _____ Жукова Н.А.

Санкт-Петербург

2020

Цель работы.

Реализовать предсказание медианной цены на дома в пригороде бостона в середине 1970-х по таким данным, как уровень преступности, ставка местного имущественного налога и т.д.

Порядок выполнения работы.

- Ознакомиться с задачей регрессии
- Изучить отличие задачи регрессии от задачи классификации
- Создать модель
- Настроить параметры обучения
- Обучить и оценить модели
- Ознакомиться с перекрестной проверкой

Требования.

1. Объяснить различия задач классификации и регрессии
2. Изучить влияние кол-ва эпох на результат обучения модели
3. Выявить точку переобучения
4. Применить перекрестную проверку по K блокам при различных K
5. Построить графики ошибки и точности во время обучения для моделей, а также усредненные графики по всем моделям

Ход работы.

В задаче классификации определяется принадлежность к одному из нескольких классов, то есть набор значений ограничен. В задаче регрессии определяется значение какой-либо характеристики объекта, значение которой может быть любое число.

Была создана и обучена модель искусственной нейронной сети в Keras, код представлен в приложении А.

При исследовании была применена перекрестная проверка по k блокам при различных k: 2, 4, 8.

Изначально была рассмотрена модель с 8 блоками, тренировавшаяся на 100 эпохах. Были выведены графики по всем 8 блокам. Графики представлены на рис. 1-8. На рисунке 9 показана усредненная ошибка по всем блокам.

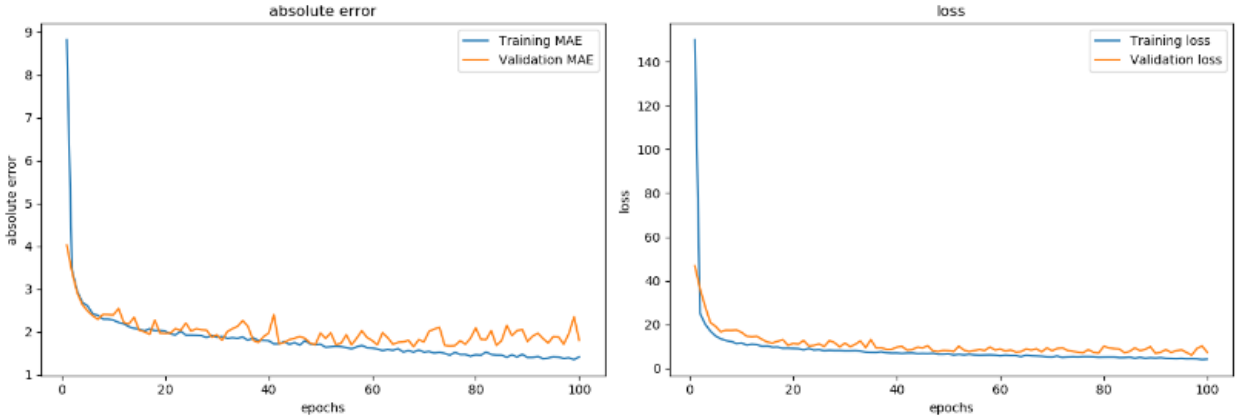

Рисунок 1 – абсолютная ошибка и потери на первом блоке.

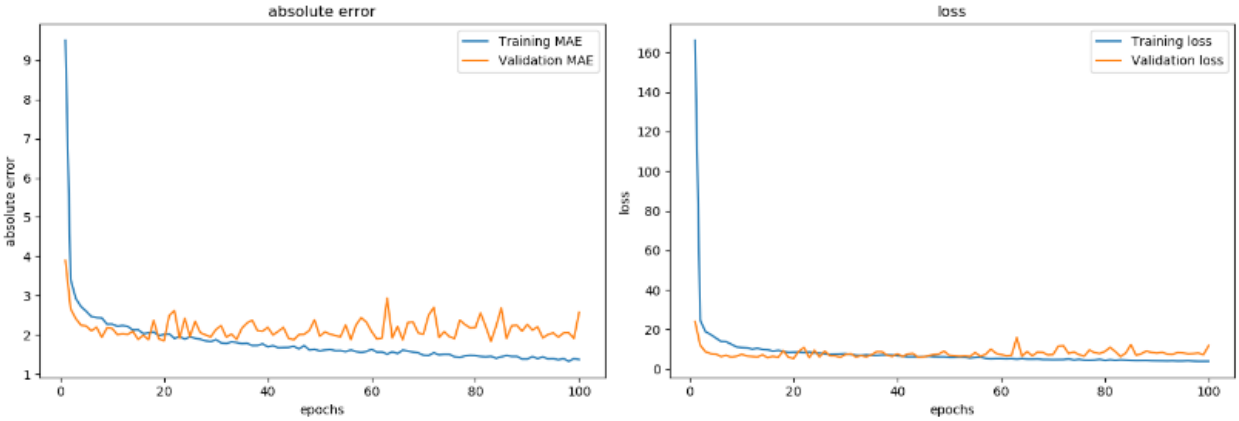

Рисунок 2 – абсолютная ошибка и потери на втором блоке.

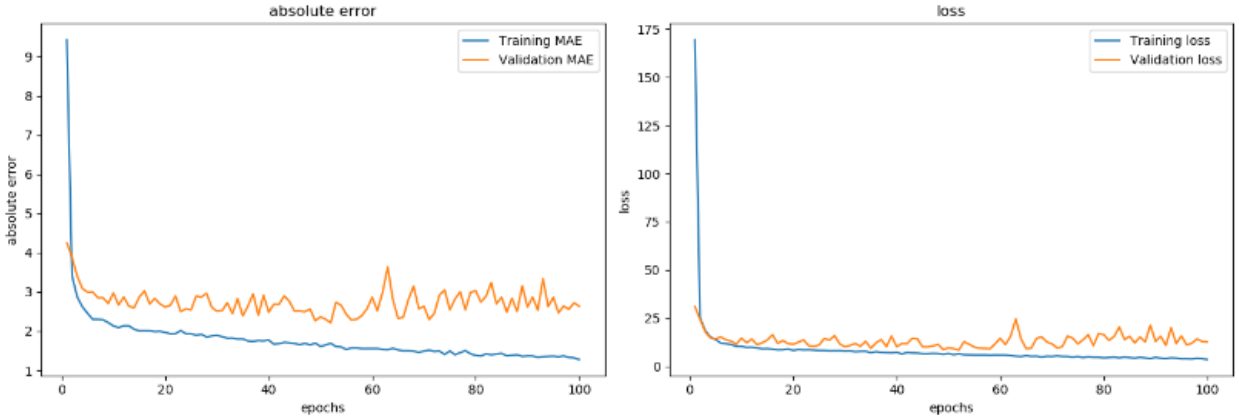

Рисунок 3 – абсолютная ошибка и потери на третьем блоке.

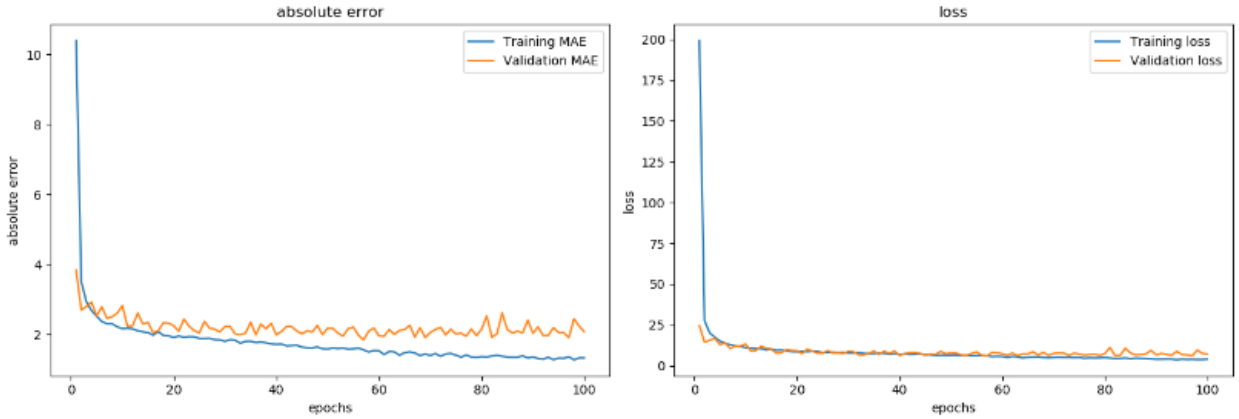

Рисунок 4 – абсолютная ошибка и потери на четвертом блоке.

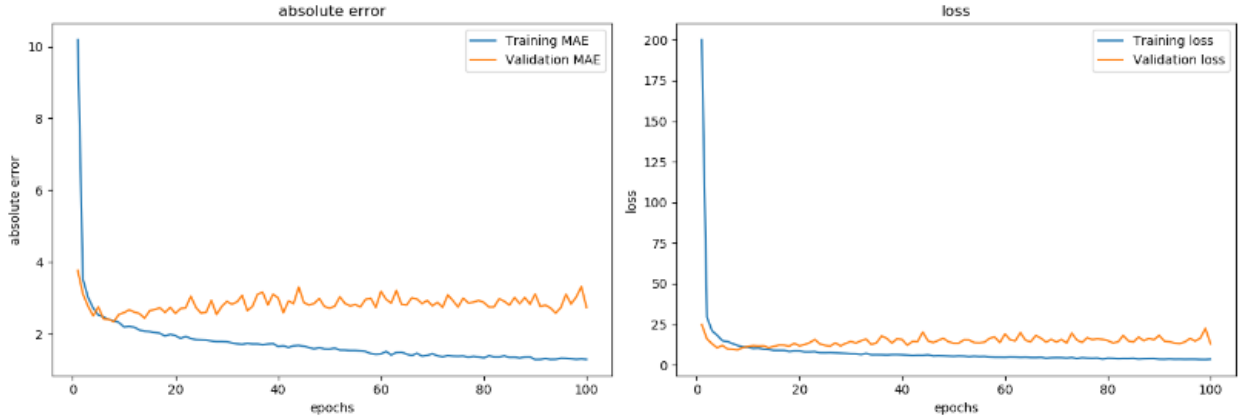

Рисунок 5 – абсолютная ошибка и потери на пятом блоке.

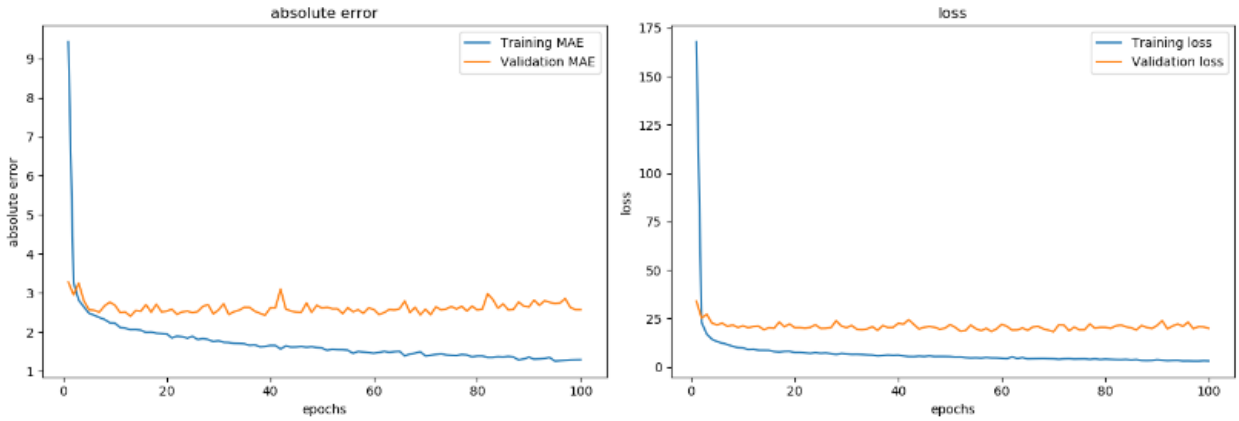

Рисунок 6 – абсолютная ошибка и потери на шестом блоке.

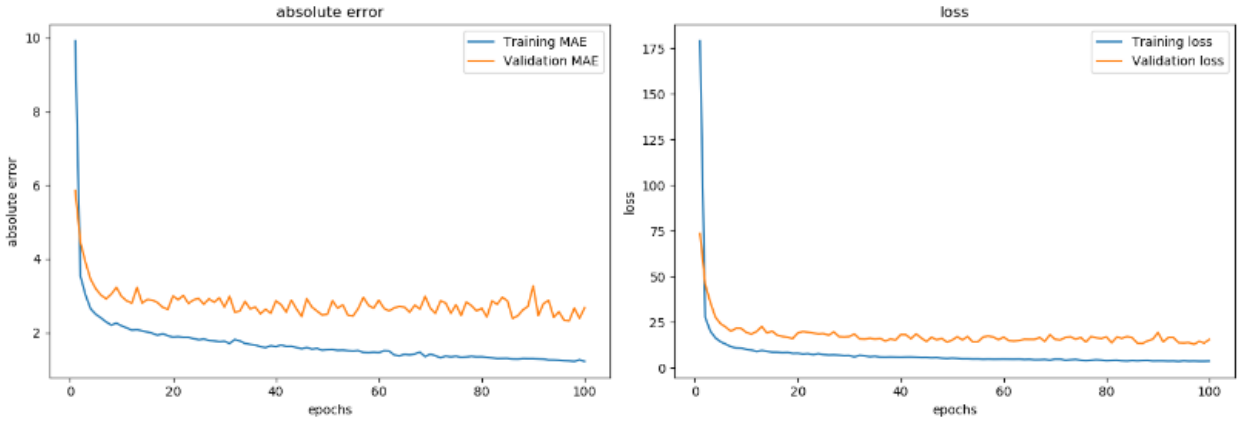

Рисунок 7 – абсолютная ошибка и потери на седьмом блоке.

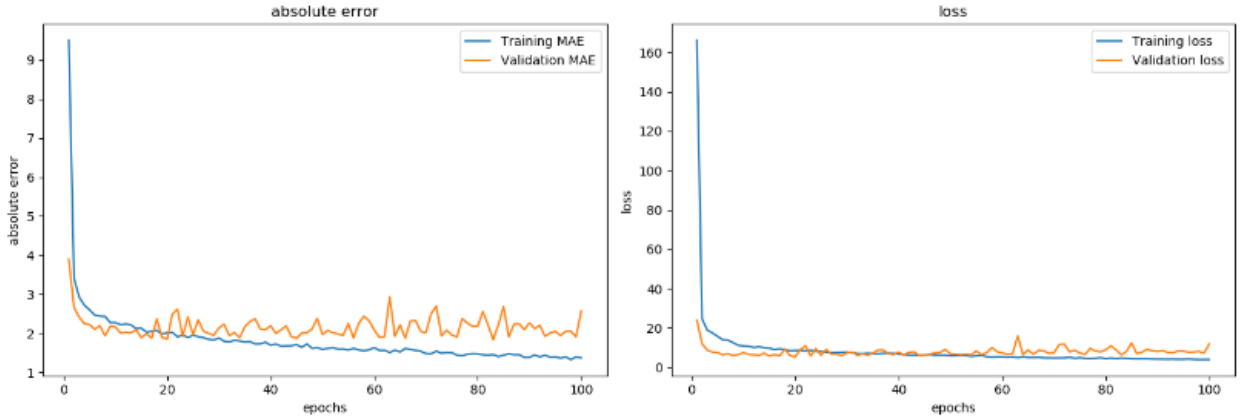

Рисунок 8 – абсолютная ошибка и потери на восьмом блоке.

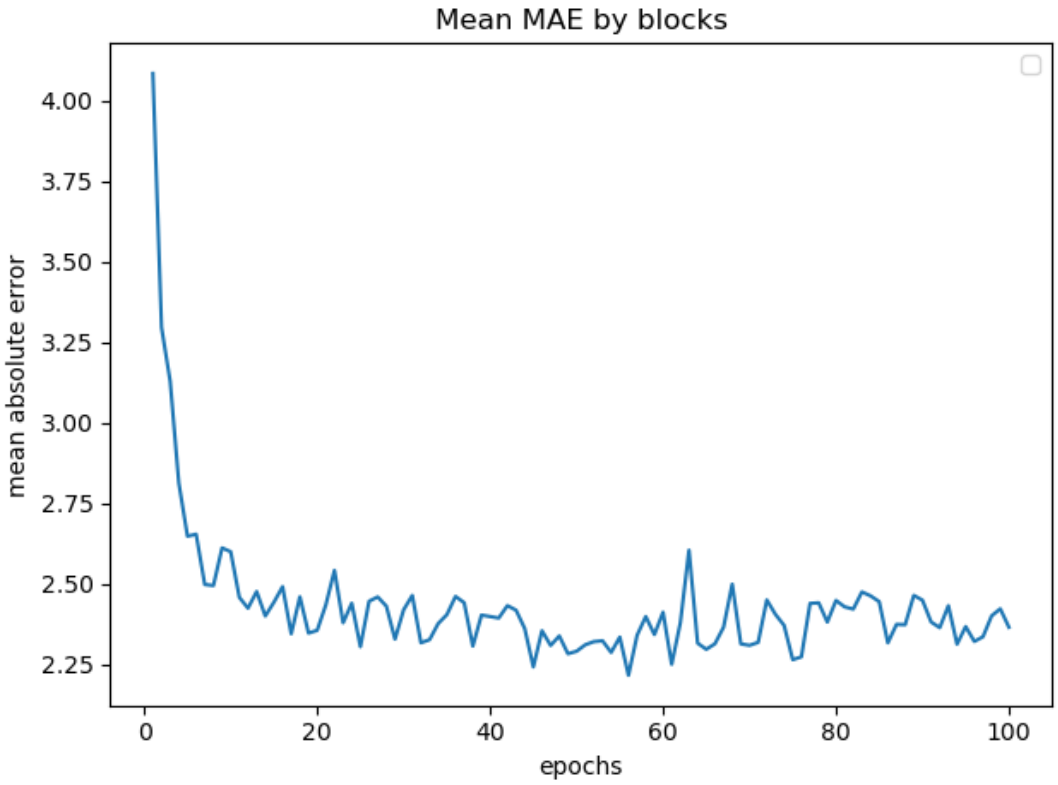

Рисунок 9 – усредненная абсолютная ошибка по всем блокам.

Как можно увидеть на рисунке 9, точка переобучения находится на ~50 эпохе (начиная с этой эпохи абсолютная ошибка не уменьшается). Далее будем тренировать сеть на 50 эпохах.

Рассмотрим модель с 4 блоками, тренировавшуюся на 50 эпохах. Были выведены графики по всем 4 блокам. Графики представлены на рис. 10-13. На рисунке 14 показана усредненная ошибка по всем блокам.

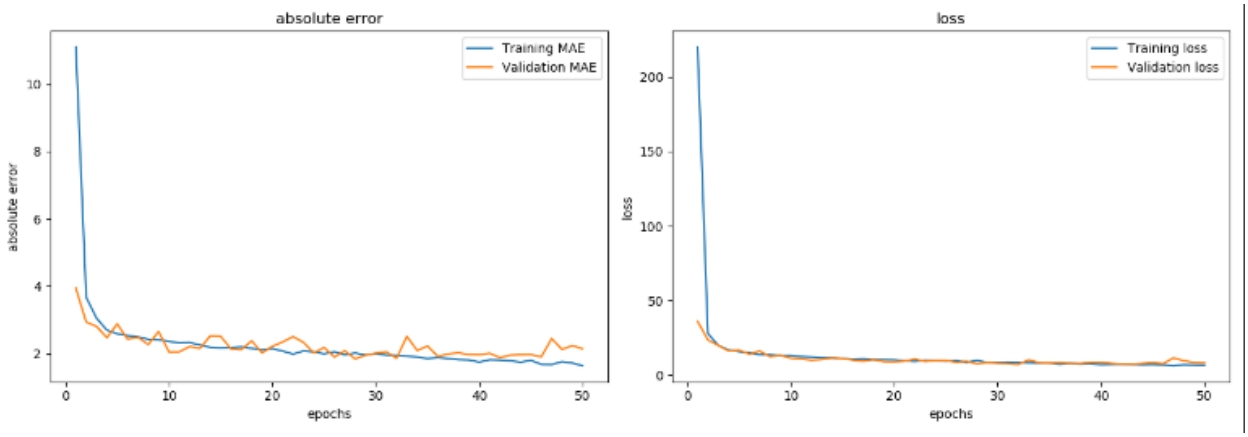

Рисунок 10 – абсолютная ошибка и потери на первом блоке.

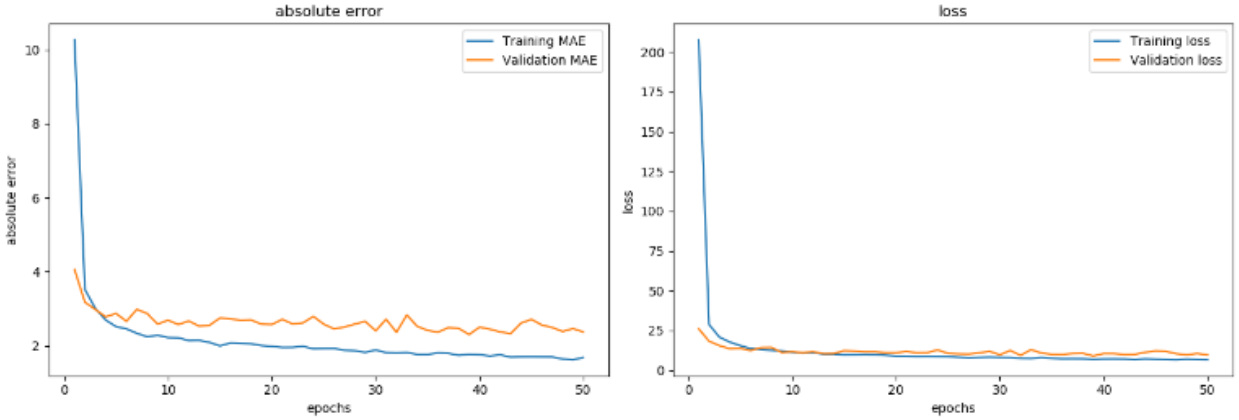

Рисунок 11 – абсолютная ошибка и потери на втором блоке.

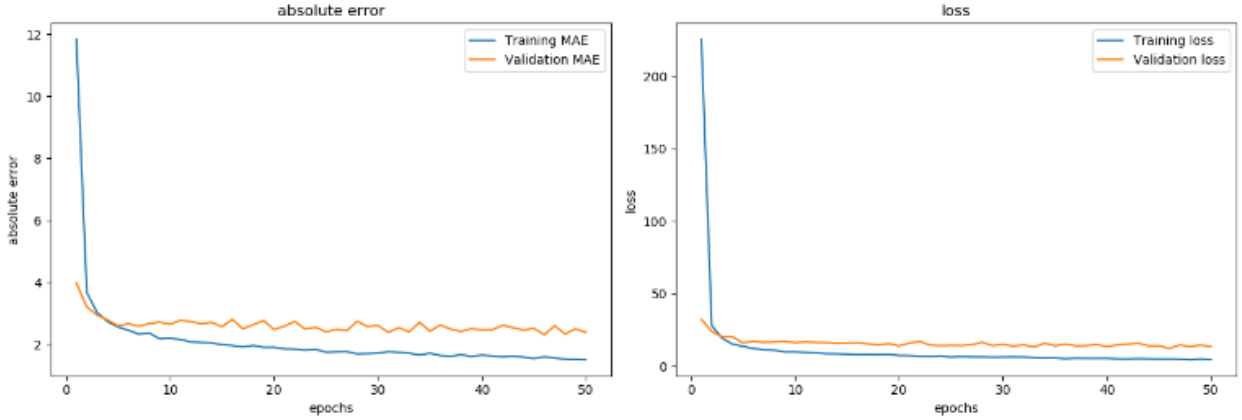

Рисунок 12 – абсолютная ошибка и потери на третьем блоке.

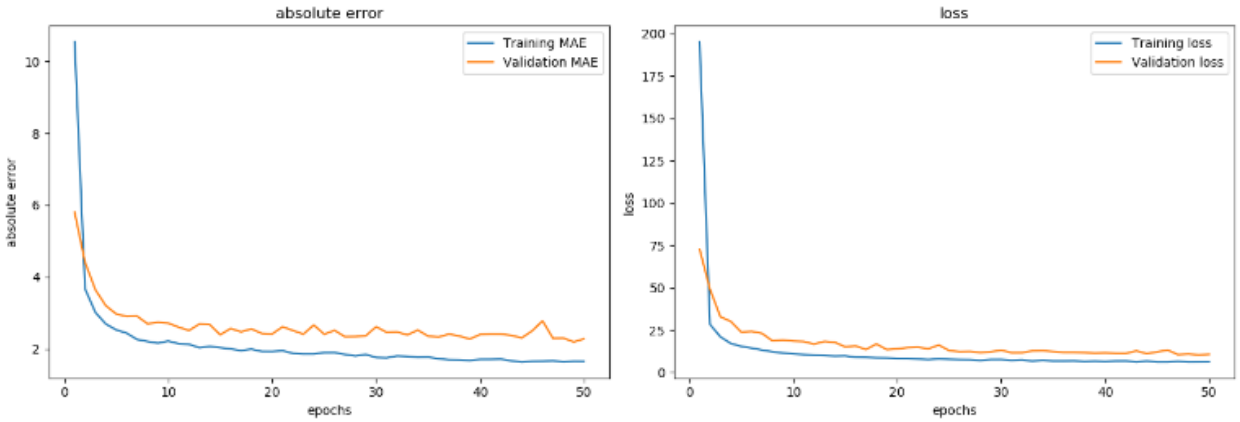

Рисунок 13 – абсолютная ошибка и потери на четвертом блоке.

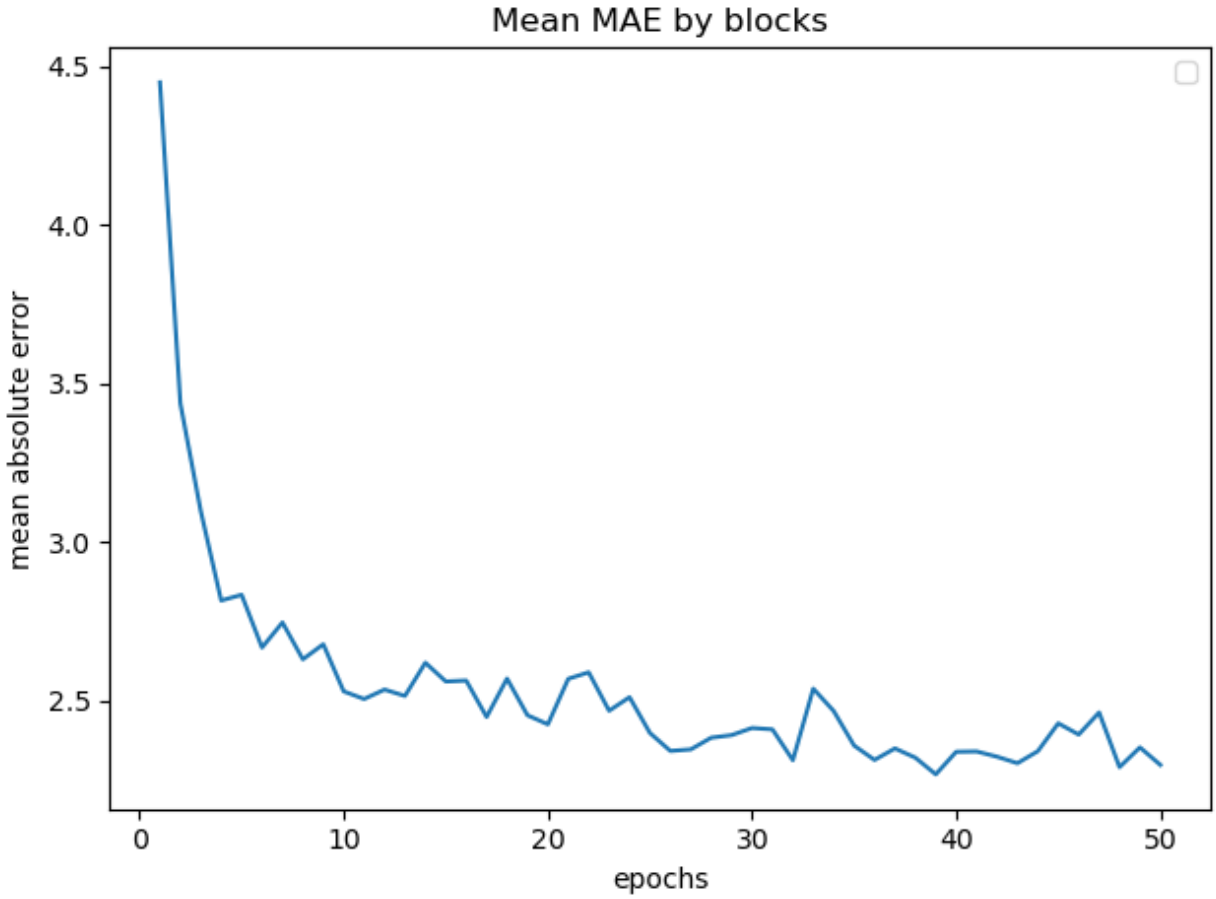

Рисунок 14 – усредненная абсолютная ошибка по всем блокам.

Рассмотрим модель с 2 блоками. Были выведены графики по всем 2 блокам. Графики представлены на рис. 15-16. На рисунке 17 показана усредненная ошибка по всем блокам.

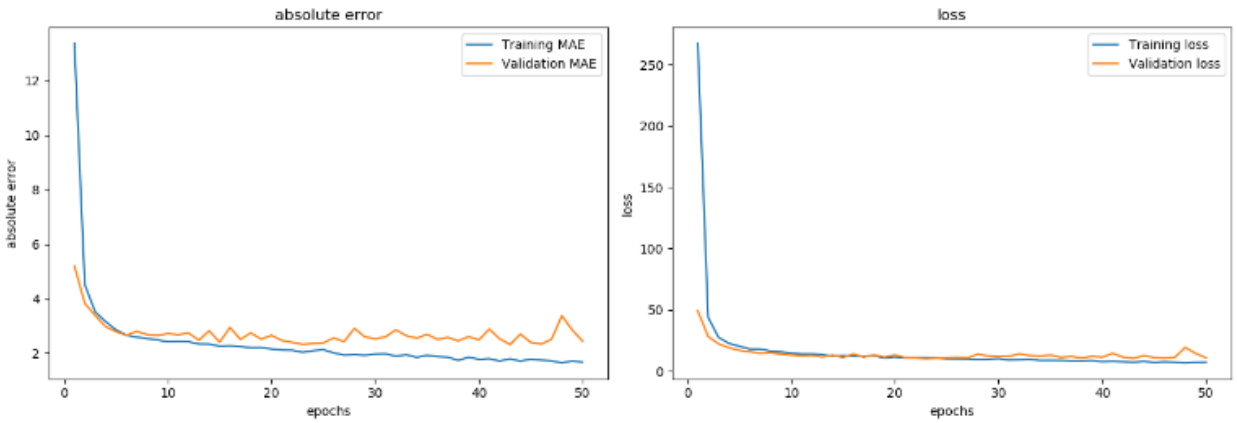

Рисунок 15 – абсолютная ошибка и потери на первом блоке.

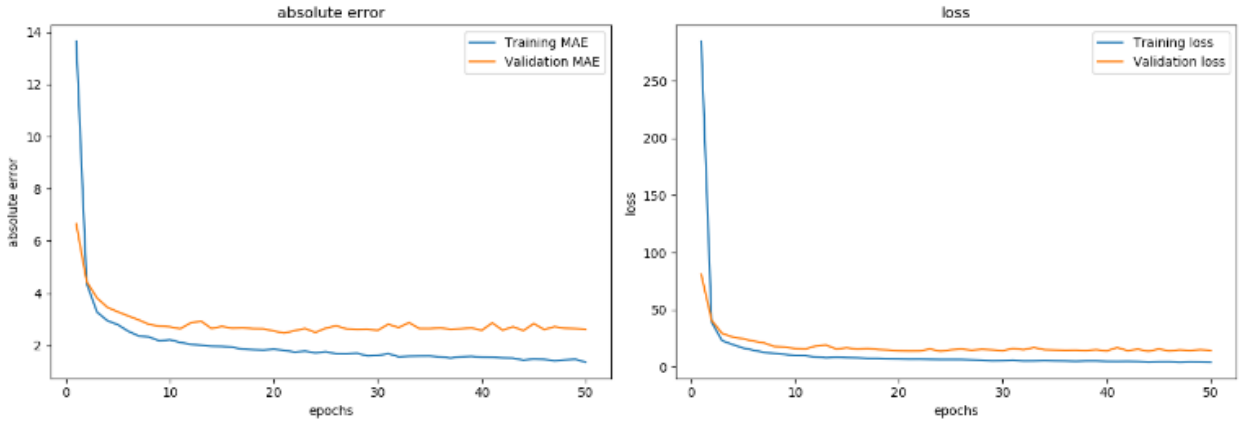

Рисунок 16 – абсолютная ошибка и потери на втором блоке.

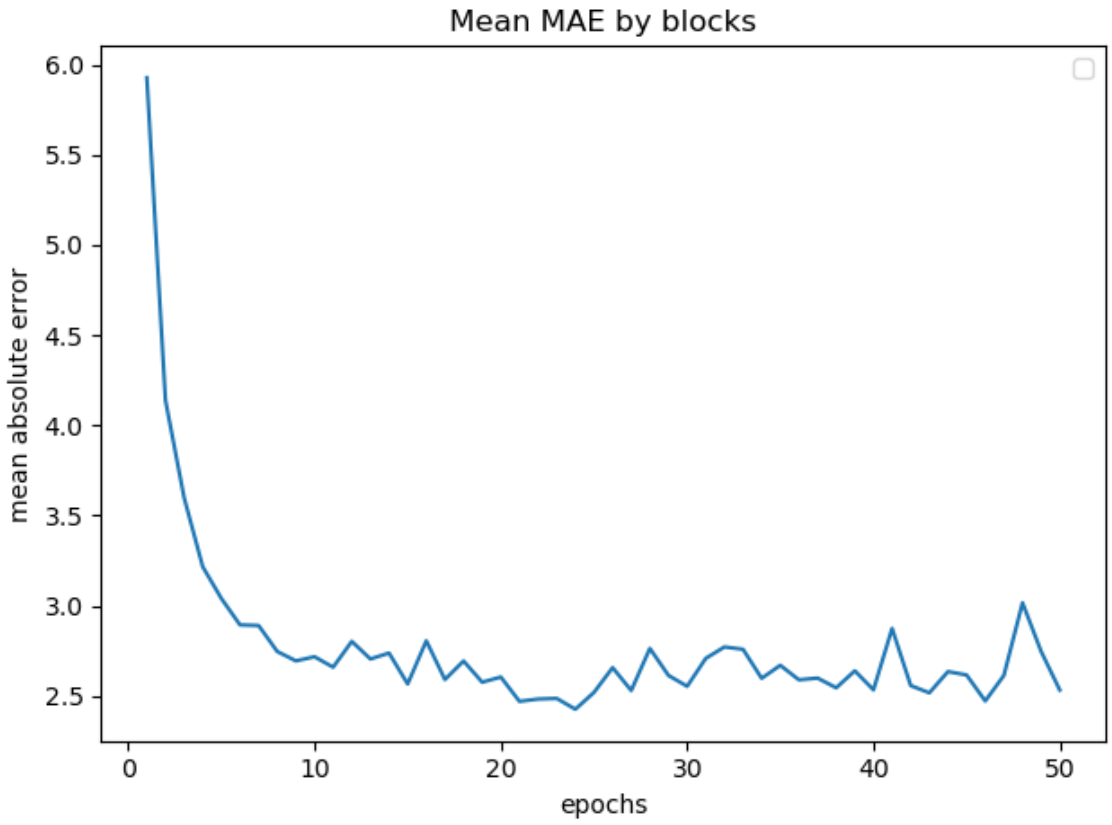

Рисунок 17 – усредненная абсолютная ошибка по всем блокам.

В среднем прогнозы отклоняются на 2531 доллара.

Выводы.

В ходе выполнения данной работы была изучена задача регрессии и ее отличие от задачи классификации. Была изучена и проведена перекрестная проверка по k блокам при различных k.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД

```
import numpy as np
import matplotlib.pyplot as plt
from keras.layers import Dense
from keras.models import Sequential
from keras.datasets import boston_housing


(train_data, train_targets), (test_data, test_targets) =
boston_housing.load_data()

print(train_data.shape)
print(test_data.shape)

print(test_targets)

mean = train_data.mean(axis=0)
train_data -= mean
std = train_data.std(axis=0)
train_data /= std

test_data -= mean
test_data /= std


def build_model():
    model = Sequential()
    model.add(Dense(64, activation='relu',
input_shape=(train_data.shape[1],)))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
    return model


k = 8
num_val_samples = len(train_data) // k
num_epochs = 50
all_scores = []

for i in range(k):
    print('processing fold #', i)
    val_data = train_data[i * num_val_samples: (i + 1) * num_val_samples]
    val_targets = train_targets[i * num_val_samples: (i + 1) *
num_val_samples]
    partial_train_data = np.concatenate([train_data[:i * num_val_samples],
train_data[(i + 1) * num_val_samples:]],
                                        axis=0)
    partial_train_targets = np.concatenate([train_targets[:i *
num_val_samples], train_targets[(i + 1) * num_val_samples:]], axis=0)
    model = build_model()
    H = model.fit(partial_train_data, partial_train_targets,
epochs=num_epochs, batch_size=1, verbose=0, validation_data=(val_data,
```

```
val_targets))
    all_scores.append(H.history['val_mae'])

    plt.figure(i + 1, (4.6 * 3, 4.8))
    plt.subplot(121)
    plt.plot(range(1, num_epochs + 1), H.history['mae'], label='Training
MAE')
    plt.plot(range(1, num_epochs + 1), H.history['val_mae'],
label='Validation MAE')
    plt.title('absolute error')
    plt.ylabel('absolute error')
    plt.xlabel('epochs')
    plt.legend()

    plt.subplot(122)
    plt.plot(range(1, num_epochs + 1), H.history['loss'], label='Training
loss')
    plt.plot(range(1, num_epochs + 1), H.history['val_loss'],
label='Validation loss')
    plt.title('loss')
    plt.ylabel('loss')
    plt.xlabel('epochs')
    plt.legend()

    plt.show()
    plt.clf()


print(np.mean(all_scores[:-1]))

average_mae_history = [np.mean([x[i] for x in all_scores]) for i in
range(num_epochs)]
plt.figure(0)
plt.plot(range(1, num_epochs + 1), average_mae_history)
plt.xlabel('epochs')
plt.ylabel("mean absolute error")
plt.title('Mean MAE by blocks')
plt.legend()
plt.show()
```